Context Forest for object class detection

Context Forest for Object Class Detection

Davide Modolo
davide.modolo@gmail.com
Alexander Vezhnevets
vezhnick@gmail.com
Vittorio Ferrari
vittoferrari@gmail.com

University of Edinburgh
Edinburgh, UK

Abstract

We present Context Forest (ConF) — a technique for predicting properties of the objects in an image based on its global appearance. Compared to standard nearest-neighbour techniques, ConF is more accurate, fast and memory efficient. We demonstrate ConF by predicting three properties: aspects of appearance, location in the image, and class membership. In extensive experiments we show that (i) ConF can automatically select which components of a multi-component detector to run on a given test image, obtaining a considerable speed-up for detectors trained from large sets (10× for EE-SVMs [36] and 2× for DPM [21]); (ii) ConF can improve object detection performance by removing false positive detections at unlikely locations (+2% mAP), and by (iii) removing false positives produced by classes unlikely to be present in the image (+5% mAP on a 200-class dataset [7]).

1 Introduction

Global image appearance carries information about properties of objects in the image. For instance, a picture of a highway taken from a car is more likely to contain cars from the back viewpoint than from the side (fig. 1) and a picture taken under water is more likely to contain a fish than a car. This shows how the global image appearance of images can help understanding what objects are present and what they look like. Moreover, another property that can be inferred from global image appearance is the rough location of object instances [45]. For instance, an urban scene with cars parked in front of a building, shows cars in the bottom half of the image (fig. 2).

In this paper we exploit this observation for object class detection. We propose Context Forest (ConF): a technique for learning the relation between the global appearance of an image and the properties of the objects it contains. Given only the global appearance of a test image, ConF retrieves a subset of training images that contain objects with similar properties. ConF is based on the Random Forest [8, 10] framework, which provides high computational efficiency and the ability to learn complex, non-linear relations between global image appearance and objects properties. It is very flexible and only requires these properties to be defined through a distance function between two object instances, e.g. their appearance similarity or difference in location. We demonstrate ConF by learning to predict three properties: aspects of appearance, location in the image, and class membership.

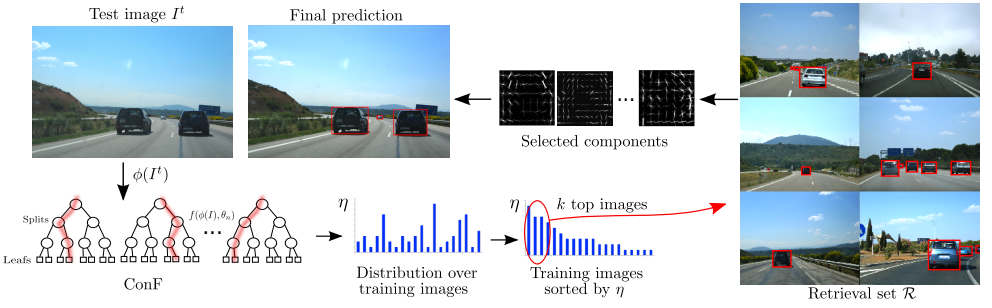


Figure 1: *Illustration of ConF selecting components for a test image (sec. 3.1). A test image I_t is passed through ConF, reaching a leaf in each tree. For each training image we then count how many trees have selected it, and we construct a retrieval set $\mathcal{R}$ that contains the k most frequently selected images. Finally, we select which components to run on I_t based on what components appear in $\mathcal{R}$.*

Multi-component detectors [4, 14, 15, 17, 21, 26, 36, 60] model an object class as a mixture of components, each trained to recognize a different aspect of appearance. For example, different viewpoints (e.g. front and back view of a car [21, 25]) or articulation states (e.g. a person sitting vs standing [52]). When trained on a large set, such a detector has many components [14, 60], which all need to be evaluated on a test image, making it slow. Instead, we use ConF to select a subset of model components which is most relevant to a particular test image. We then run only those components, obtaining a speed-up (2x for DPM [21] and 10x for EE-SVM [36], sec. 4.3). Hence, ConF makes large multi-component detectors practical. This is particularly useful for EE-SVMs, as they have as many components as there are training instances. Interestingly, in some cases we even gain a small improvement in accuracy, by not running some components that would produce false positive detections.

Moreover, we train a second ConF to predict at which positions and scales objects are likely to appear in a given test image, analogue to [33, 45, 55]. By incorporating this information in the detector score at test time, we reduce the false positive rate by removing detections at unlikely locations. Experiments show an mAP improvement of 2% (sec. 4.4).

Finally, we train a third ConF to predict what object classes are present in an image, as in [1, 27, 51]. We use it at test time by only running detectors of classes predicted to be in the test image. In experiments on a 200-class dataset [2], this enables to run just 10 detectors per image on average, while also improving mAP by 5%, as it removes false positives produced by detectors of classes unlikely to be present in the image (sec. 4.5).

All these experiments demonstrate that ConF is a general technique that can predict various kinds of object properties. We carry out an extensive comparison to standard nearest-neighbour techniques for such context-based predictions [33, 42, 45, 53, 55]. It shows that ConF predicts object properties from global image appearance more accurately, while being much faster and memory efficient (sec. from 4.2 to sec. 4.6).

2 Related work

Context. The use of context for object detection is a broad research area. Some works [9, 13, 21, 28, 42] model context as the interactions between multiple object class detectors in the same image. In this paper, we model context as the relation between global image appearance and properties of the objects within them, as in [27, 33, 39, 45, 51, 53, 55, 59]. These works have shown that global image descriptors give a valuable cue about which classes might be present in an image and where they are located. Many object detectors [1,

27, 39, 45, 51, 55] employed such global context to remove out-of-context false-positive detections. A similar approach was also used for image parsing [33, 53, 59]. Most of these works have a nearest neighbour core: they first retrieve a small subset of training images which are most globally similar to a test image, and then transfer the relevant statistics of the object properties in this retrieval set to the test image. In our work instead the retrieval set is estimated by ConF, which is *explicitly trained* to return images containing objects with similar properties to those in the test image. ConF has several advantages over nearest-neighbour approaches: (i) it can learn highly complex non-linear dependencies between the global descriptor and the object property. As a result, it estimates it more accurately; (ii) in large training sets, nearest neighbour becomes very slow, as its complexity is linear in their size. ConF is much faster and more memory efficient; (iii) ConF supports any objective function, which might even be evaluated on a different data representation than the input at test time. This is a crucial feature for our problem, as we want to predict properties of objects, but based on global image features.

Multi-component detectors. These detectors [4, 14, 15, 17, 21, 26, 36, 60] model an object class as mixture of components. They can be slow when trained from large training sets as they need many components to reach peak performance [14, 60]. While we present experiments on DPM [21] and EE-SVM [36], ConF can in principle speed-up any multi-component detector. The problem of speeding-up detection, especially when evaluating many HOG templates, has also been attacked by other works [11, 20, 46, 49]. However, these are specialized to the HOG/DPM detectors, as they exploit its internal structure. Moreover, [11, 49] are multi-class methods and achieve a speed-up only when the number of classes predicted at the same time is large. ConF, instead can speed-up even a single class (sec. 4.3). Finally, we believe ConF can offer an additional, complementary way to speed-up, and could potentially be combined with these works.

EE-SVM. The EE-SVM [36] is an extreme case of multi-component detector, where a separate component is created for each training example. Because of this, EE-SVM benefit the most from dynamically selecting components with ConF. EE-SVM is widely used for applications beyond object detection [5, 6, 16, 18, 24, 30, 37, 47, 48, 50, 54, 57] because it explicitly associates a training example to each object it detects in a test image. This enables transferring meta-data such as segmentation masks [36, 54], 3D models [36], viewpoints [5], GPS locations [24] and part-level regularization [6]. Furthermore, EE-SVM can also be used for discovering objects parts [18, 48], scene classification [30, 48], object classification [16], image parsing [54], image matching [47], automatic image annotation [57] and 3D object detection [50]. All these applications can potentially be accelerated by ConF.

3 Estimating object properties from context

We exploit the observation that global image appearance contains information about properties of the objects inside it. We focus on three properties: aspect, location, and class. We propose a new method (ConF) based on the Random Forest framework [8, 10], which learns the relation between global image features and the properties of the object in that image. Given only the global image appearance of a test image, ConF retrieves a subset of training images that contain objects with similar properties. The properties in this retrieval set can then be used to modify the behaviour of the object detector, e.g. by running only some components or by downgrading the score of false positive detections at unlikely locations. In the following, we first describe ConF in its general form (sec. 3.1), and then specialize it for each of the three properties we consider (sec. 3.2 to 3.4).

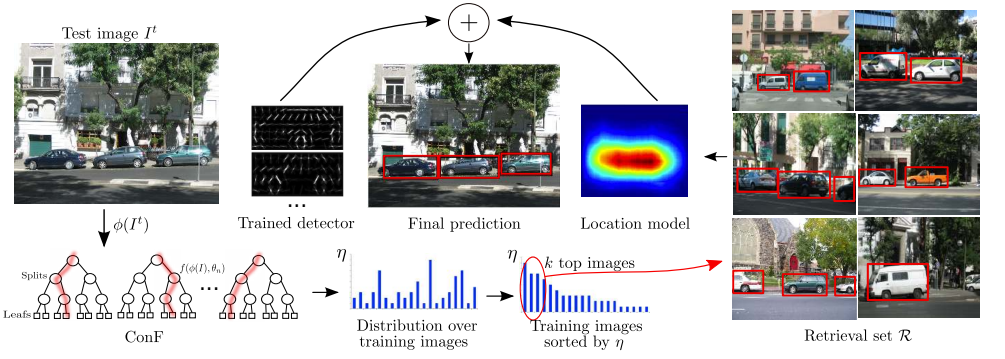


Figure 2: *Illustration of ConF predicting object locations (sec. 3.3).*

3.1 Context Forest (ConF)

Given a training set $\mathcal{T}$ the goal of ConF is to map the global appearance $\phi(I_t)$ of a test image I_t into a retrieval set $\mathcal{R} \subset \mathcal{T}$. We want to construct a mapping, such that properties of objects in images of $\mathcal{R}$ are similar to the properties of objects in I_t (e.g. appearance, location, class).

ConF at training time learns an ensemble of decision trees (forest) that operates on global image features $\phi(I)$. We construct each tree by recursively splitting the training set $\mathcal{T}$ at each node. We want the leaves of the trees to contain images whose objects properties are compact according to some measure $c(\mathcal{T}_l)$, where $\mathcal{T}_l$ are the training images in leaf l. Each internal node n contains a binary split function $f(\phi(I),\theta_n)$, where θ_n are its parameters. Let $\mathcal{T}_n$ be the training images that reached node n, then $f(\phi(I),\theta_n)$ will split $\mathcal{T}_n$ into two subsets $\mathcal{T}_l$ and $\mathcal{T}_r$. We use axis-aligned weak learners as f [10]. The split function $f(\phi(I),\theta_n)$ applies a threshold to one of the dimensions of the image feature vector $\phi(I)$. Following the extremely randomized forest approach [33], for each node we randomly sample several thousand possible splits θ and choose the one that maximizes the joint compactness:

$$\theta_n = \arg\max_{\theta} c(\mathcal{T}_l) + c(\mathcal{T}_r) \tag{1}$$
$$\text{s.t. } \forall I \in \mathcal{T}_l,\ f(\phi(I),\theta) = 0,\ \forall I \in \mathcal{T}_r,\ f(\phi(I),\theta) = 1$$

where compactness is defined as

$$c(\mathcal{T}) = \frac{1}{N(\mathcal{T})^2} \frac{1}{\sigma^2\sqrt{2\pi}} \sum_{w_i \in \mathcal{T}} \sum_{w_j \in \{\mathcal{T} \setminus w_i\}} e^{-\frac{1}{2}\frac{D(w_i,w_j)^2}{\sigma^2}} \tag{2}$$

where $N(\mathcal{T})$ is the number of ground-truth object bounding-boxes in $\mathcal{T}$ and $D(w_i,w_j)$ is a distance measure between the properties of two object bounding-boxes w_i and w_j. Note how the inner summation in (2) is an estimation of the density of the distribution induced by all bounding-boxes in $\{\mathcal{T} \setminus w_i\}$, evaluated at w_j. This value is high if w_j has other bounding-boxes nearby. The estimate is done with a Gaussian Kernel Density estimator [21] (KDE). We estimate the standard deviation σ from the entire training set once before training the forest. This determines the scale of the problem, i.e. at which range two bounding-boxes should be considered close. For each training bounding-box w_i we compute its k-nearest neighbours in the whole training set and compute the standard deviation over them. Finally, we set σ as the median of these standard deviations over all bounding-boxes.

By employing different compactness measures c, we can use ConF to learn relations between different object properties and global image features. Later we show how to use it for

selecting components relevant for a test image (sec. 3.2), for estimating likely object locations in it (sec. 3.3), and for predicting which object classes it is likely to contain (sec. 3.4).

ConF at test time operates in two phases (fig. 1, 2 and 3). First, test image I_t is passed through the forest, reaching a leaf in each tree. Thereby, each tree selects the subset of training images contained in that leaf. For each training image I_i, we count how many trees have selected it, forming the score $\eta(I_i, I_t)$. We now construct the retrieval set $\mathcal{R}$ to contain the k most frequently selected training images. In our experiments $k = 20$.

Note how the split function f and the compactness measure c operate in structurally different spaces. While f operates on the global image features $\phi(I)$, c is measuring the similarity of properties of *objects inside the images*. In this way ConF learns the relation between the two. Importantly, c is neither convex nor differentiable in $\phi(I)$ and we are only able to learn this inter-space relation thanks to the unique advantages of Random Forests.

3.2 ConF for component selection

Here we assume that each component of an object detector has been previously trained from a visually compact set of object instances, and that we know the component id ξ_j of each training instance j. In EE-SVMs each training instance leads to a different component. In DPM the component id of a training instance can be inferred by the output of the training procedure [22]. Based on this information, we train a ConF to select a small subset of components to run on a given test image I_t.

Training. To train ConF for components selection we define the distance $D(w_i, w_j)$ in (2) as the L2 distance between the HOG descriptors of bounding-boxes w_i and w_j.

Test. We pass the test image I_t through ConF obtaining a retrieval set $\mathcal{R}$. We then estimate a posterior distribution $p(\xi_j|I_t)$ over components ξ_j given I_t. As a training image I might contain multiple instances from different components, each training image is 'labelled' by a distribution over components $p(\xi_j|I)$. We estimate the component distribution for the test image I_t as the average over the training images in the retrieval set $\mathcal{R}$

$$p(\xi_j|I_t) = \frac{1}{|\mathcal{R}|} \sum_{I \in \mathcal{R}} p(\xi_j|I) \tag{3}$$

Based on this distribution, we can now select which components to run on I_t. We rank components by their probability and iteratively pick them until their combined probability mass exceeds a threshold γ. This threshold controls a trade-off between running few components and getting high detection performance. An interesting aspect of our formulation is that the number of selected components changes depending on the test image. A test image with a characteristic appearance matching training images with a systematic recurrence of a few components will lead to a peaky $p(\xi_j|I_t)$. In this case it is safe to run only a few components and we obtain a substantial speed-up. On the other hand, if the ConF is uncertain about the contents of the test image, then the entropy of $p(\xi_j|I_t)$ will be high, and many components will be selected. In the extreme case, for a very difficult test image, our procedure naturally degenerates to the default case of running all components (assuming a high threshold γ).

3.3 ConF for object location

At test time, a typical detector scores windows in the test image I_t, based on their appearance only. We propose here to augment the detector's scores by adding knowledge about likely positions and scales of the object class, derived purely from the global appearance of I_t.

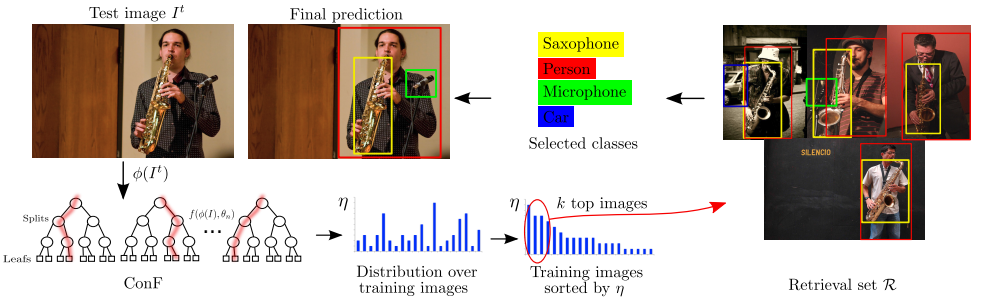


Figure 3: *Illustration of ConF predicting class membership (sec. 3.4).*

Training. We train two ConFs to predict object positions and scales, respectively. To do so, we employ a different measure of compactness, substituting the distance between two bounding-boxes in (2) with D_{POS} (or D_{SCALE}). We define $D_{\text{POS}}(w_i, w_j)$ as the L2 distance between w_i and w_j. We define $D_{\text{SCALE}}(w_i, w_j) = \max(\frac{H_i}{H_j}, \frac{H_j}{H_i}) \cdot \max(\frac{W_i}{W_j}, \frac{W_j}{W_i})$ as the difference in their scale (W and H refer to width and height).

Test. We first pass the test image I_t through ConF obtaining a retrieval set $\mathcal{R}$, and then compute the following score for each window w in the test image:

$$\frac{1}{N(\mathcal{R})} \sum_{w_i \in \mathcal{R}} \frac{1}{\sigma^2 \sqrt{2\pi}} e^{-\frac{1}{2} \frac{D(w,w_i)^2}{\sigma^2}} \tag{4}$$

where $N(\mathcal{R})$ is the number of object instances in $\mathcal{R}$, and D is either D_{POS} or D_{SCALE} . We learn σ from the entire training set as in sec. 3.1. This score captures how likely a window is to cover an object based on its position/scale. Finally, we linearly combine the position/scale scores with the detector's score of w. The usual non-maxima suppression stage follows.

3.4 ConF for class selection

In multi-class problems, a typical system would run detectors for all classes on all test images [43]. Instead, here we use ConF to predict what classes are present in each image, and run only the corresponding detectors. This greatly reduces the number of detectors run, and removes some false-positives.

Training. To train ConF to predict what classes are present in an image, we use the following distance between two bounding-boxes in the compactness function (2):

$$D_{\text{CLASS}}(w_i, w_j) = \begin{cases} 0 & \text{if } w_i \text{ and } w_j \text{ are objects of the same class} \\ \infty & \text{otherwise} \end{cases} \tag{5}$$

This definition simplifies (2) considerably, as $e^{-\frac{1}{2} \frac{D(w_i,w_j)^2}{\sigma^2}}$ can only be 0 or 1. Note how the inner summation in (2) now simply counts the number of objects $w_j \in \{\mathcal{T} \backslash w_i\}$ of the same class as w_i. We can rewrite (2) equivalently as:

$$c(\mathcal{T}) = \frac{1}{N(\mathcal{T})^2} \frac{1}{\sigma^2 \sqrt{2\pi}} \sum_{c \in C} N(\mathcal{T}, c) \cdot (N(\mathcal{T}, c) - 1) \tag{6}$$

where $N(\mathcal{T}, c)$ counts how many objects in $\mathcal{T}$ belong to class c. Note how (6) allows an important speed up over (2), as it avoids computing the KDE. Evaluating (6) takes time $O(|\mathcal{T}|)$, compared to $O(|\mathcal{T}|^2)$ for (2).

Test. We pass the test image I_t through ConF obtaining a retrieval set $\mathcal{R}$. We then estimate a posterior distribution $p(c_j|I_t)$ over object classes c_j given I_t, analog to what done for components of one class in sec. 3.2. Based on this distribution, we can now select which detectors to run on I_t. In our experiments we run all detectors with $p(c_j|I_t) > 0$.

4 Experiments and conclusions

We present a series of experiments on two datasets (sec. 4.1). In sec. 4.2 we first evaluate how good the retrieval sets generated by ConF are. Then, we show how ConF for component selection (sec. 4.3), object location (sec. 4.4) and class selection (sec. 4.5) can be used to improve object detection. Finally, we show how ConF is more computation and memory efficient then a nearest neighbour approach (sec. 4.6).

4.1 Datasets

We present experiments on two datasets: a 2-class dataset we call BigCH, and the 200-class `val` subset of ILSVRC2014 [7]. The first one has few classes, but many training instances per class (on average $\sim 21k$). We use it to evaluate ConF for selecting model components (sec. 4.3) and predicting object location (sec. 4.4). The second dataset has many classes, but much fewer training instances per class (on average ~ 530). We use it to evaluate ConF for class selection (sec. 4.5).

BigCH is a 2-class dataset (*Car* and *Horse*). It combines 6 existing datasets: PASCAL VOC 2012 [19], ImageNet [12], LabelMe [44], SUN 2012 [58], UIUC [3] and PASCAL-10x [60]. We collected all images with bounding-box annotations and removed duplicate images and incorrect bounding-boxes. In total, the dataset has 15766 images containing 28548 car instances and 10107 images containing 13071 horse instances. Finally, we collected negative images so that each source dataset contributes an equal number of positive and negative images. We split the dataset into training (90%) and test (10%) sets. The training set contains 25774 cars and 9097 horses, and the test set contains 2774 cars and 1010 horses.

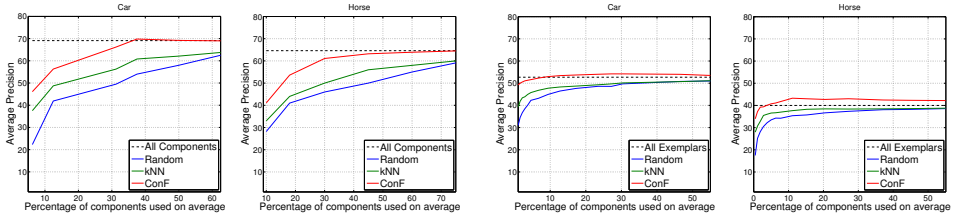
ILSVRC2014 [7] is a 200-class dataset annotated by bounding-boxes. Following [23], we perform experiments on the `val` set and consider two disjoint subsets: $\texttt{val}_1$ (10k images) and $\texttt{val}_2$ (10k images). We use $\texttt{val}_1$ for training and test on $\texttt{val}_2$.

4.2 Quality of retrieval sets

We quantify how similar object bounding-boxes from a test image I_t are to those in the retrieval set $\mathcal{R}$ returned by the method using the average density of $\mathcal{R}$ evaluated at the object properties in I_t

$$\frac{1}{Z\sigma^2\sqrt{2\pi}} \sum_{w_i \in I_t} \sum_{w_j \in \mathcal{R}(I_t)} e^{-\frac{1}{2}\frac{D(w_i,w_j)^2}{\sigma^2}} \tag{7}$$

where Z is the number of pairs of bounding-boxes in I_t and $\mathcal{R}$. The distance D and standard deviation σ vary depending on the property, as defined in sec. 3.2-3.4. We extract different features as global image descriptors $\phi(I)$, depending on the property. For the appearance and location properties, we extract SURF [7], LAB and SIFT [34] features on a dense grid at multiple scales. For each feature type and object class we train a codebook of 1000 visual words and construct a 2-level spatial pyramid [32]. Additionally, we also extract a GIST [40] descriptor for the image. Overall, we train ConF on a 16000 dimensional feature space. For the class membership property, we extract state-of-the-art convolutional neural network (CNN) descriptors of 4096 dimensions [29]. These are the output of a CNN pre-trained specifically for image classification [31] and so they are very suited to the task.



(a) DPM **(b) EESVM**

Figure 4: *Results of applying ConF for the automatic component selection. The points on the plot correspond to different choices for the threshold γ (sec. 3.2). The horizontal axis is the average amount of components used. The vertical axis is the AP of the detector using components selected by ConF.*

Table 1 show results averaged over the test set, higher values are better. For appearance and location, we also average results over the two classes (car and horse). As a baseline, we return the whole training set as the retrieval set. This leads to a generic prior on image properties, independent of the test image. Moreover, we compare to the traditional way of building retrieval sets by k-nearest neighbours (kNN) [33, 42, 45, 53, 55], defined on the same features as ConF. Both kNN and ConF greatly outperform the baseline, proving they return meaningful retrieval sets. This confirms the observation that the global image appearance conveys information about the objects properties inside the images. Moreover, ConF returns better retrieval sets than kNN across all object properties and retrieval set sizes evaluated. In the next sections we show how more accurate retrieval sets result in better detection performance and greater speed ups.

Obj property	All train data	NN		ConF	
		1	10	1	10
Appearance	0.02	0.07	0.03	**0.09**	0.07
Position	0.36	0.90	0.70	**1.20**	1.00
Scale	0.04	0.07	0.06	**0.09**	0.08
Class	0.05	0.36	0.30	**0.39**	0.32

Table 1: *Evaluation of the quality of retrieval sets for predicting object properties. Each entry represents the average density of the retrieval set $\mathcal{R}$ evaluated at the objects properties in the test images. We consider two sizes for $\mathcal{R}$: 1 and 10.*

4.3 ConF for automatic component selection

We now use ConF to select object detector components relevant for a given test image on the BigCH dataset. We present experiments on two detectors: DPM [21] and EE-SVM [36].

DPM trains a mixture of components, each on a subset of the data with compact HOG appearance [14, 21]. We use the publicly available implementation [22] and train 16 components for the class 'car' and 10 components for 'horse' (these number of components lead to best performance on the large BigCH dataset).

The EE-SVM model is composed of a separate linear SVM for every training instance (exemplar), trained using the exemplar as the only positive against all negatives in the training set (here on HOG features). We refer to a single exemplar SVM (E-SVM) as a component of the EE-SVM model, by analogy with DPM components. We use the publicly available implementation [35].

Fig. 4 shows the evolution of AP while increasing the percentage of components used (higher is better). We compare to building retrieval sets by kNN, and to a baseline which randomly selects components without looking at the test image. ConF outperforms the baseline and kNN for both object classes, for both detectors, and over the whole range of the plots. By employing ConF, we closely match the performance of the full DPM model by running roughly half of the components. We match the performance of a full EE-SVM when running

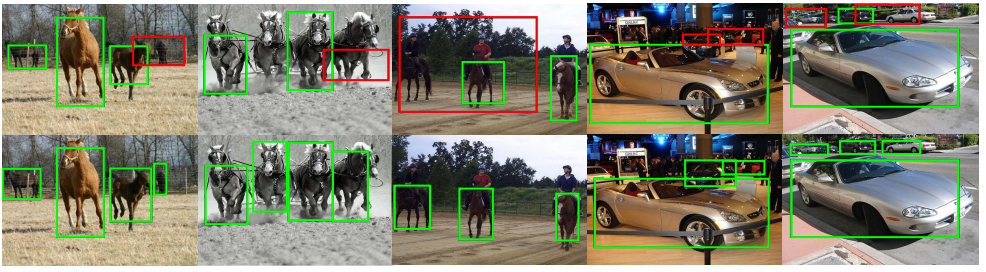


Figure 5: *Detection obtained before (top row) and after (bottom row) applying ConF for component selection. Green bounding-boxes highlight correct detections, while red ones show false positives.*

less than 10% of the components. Even in the extreme case of running just *one* EE-SVM component, the AP is about 90% of that of the full model. Interestingly, for EE-SVM on the horse class, ConF *improves AP by 3%* over the full ensemble using all components, when running *10× fewer components*. There is also a minor improvement in AP for EE-SVM on the car class. This comes from dropping some components that lead to false positives.

These experiments demonstrate the ability of ConF to select relevant components given just global image appearance. This makes EE-SVMs practical even when trained from large sets with tens of thousands of exemplars. This is crucial for applications that exploits the useful ability of EE-SVM to associate training exemplars to objects detected in test images. Fig. 5 shows some example results.

4.4 ConF for object locations

Here we demonstrate how ConF trained to estimate the location of objects from global image features can improve detection performance by downgrading the score of false positives at unlikely locations. We experiment with DPM and EE-SVM on BigCH, as in sec. 4.3. As tab. 2 shows, ConF improves AP for both classes and both detectors (+2% for cars and +1% for horses). Instead, kNN does not bring any improvement. Fig. 6 shows example results.

DPM - *Car*			EE-SVM - *Car*		
None	NN	ConF	None	NN	ConF
69.1	0	+2.1	52.7	0	+2.3
DPM - *Horse*			EE-SVM - *Horse*		
None	NN	ConF	None	NN	ConF
64.6	0	+0.7	40	0	+1.1

Table 2: *Results of augmenting the detector score with the location model derived by ConF (sec. 3.3) and NN compared to not using location model at all.*

4.5 ConF for class selection

Here we experiment on the ILSVRC2014 dataset, which has a large number of classes (200). In this scenario, we can use ConF to to predict what classes are present in an image, and then only run detectors for those classes (sec. 3.4). We use the EE-SVM detector, but this time based on object proposals [56] and state-of-the-art R-CNN features [23]. These are produced by CNN pre-trained for image classification [29, 31] and then fine-tuned for object detection [23]. For training an E-SVM, we set $C = 10^{-3}$ and we mine hard negatives from 2000 random training images (using more did not bring any improvement). Both fine-tuning and E-SVM training are done on the $\texttt{val}_1$ set. We measure test performance on $\texttt{val}_2$, as AP averaged over the 200 classes (mAP).

Without any context, EE-SVM achieves an mAP of 16.3%. Selecting classes based on kNN retrieval sets improves performance by +3.3% (mAP 19.6%), while ConF delivers a larger improvement of +4.8% (mAP 21.1%). The improvements are due to removing false positives produced by detectors of classes unlikely to be present in the image. Interestingly, ConF selects less than 10 classes per image on average, and therefore runs 20× fewer EE-SVM detectors than the context-free baseline.

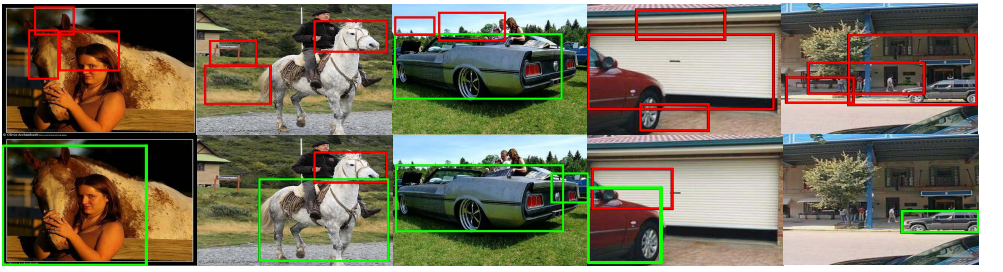


Figure 6: *Detection obtained before (top row) and after (bottom row) applying ConF as location model. Green bounding-boxes highlight correct detections, while red ones show false positives.*

4.6 Computational and memory efficiency

ConF does not only offer better performance than kNN, but is also more memory and computationally efficient. kNN requires a number of distances computations linear in the number of training images, whereas ConF requires only a logarithmic number of thresholding operations. In practice this makes a big difference in runtime, e.g. ConF for component selection takes on average 0.5s per image vs 9.9s for kNN (on a 4-cores Intel Core i5 2.0GHz).

In terms of memory, kNN stores all feature vectors of all images in the training set. For cars in BigCH, this amounts to 1.7 GB. For each internal node ConF stores a threshold, a feature id and the ids of its children, amounting to 16 bytes. The leaves store the indices of the training images they contain, for a total of exactly the number of training images $\times 2$ bytes overall (per tree). For cars in BigCH, there are < 900 internal nodes on average per tree. As we store 750 trees per class, the grand total is only 27 MB ($60\times$ less than kNN).

4.7 Conclusions

We presented Context Forest (ConF), a technique for predicting properties of objects in an image based on its global appearance. We trained ConF to predict three properties: aspects of appearance, location in the image, and class membership. We presented and extensive comparison to standard nearest-neighbour techniques for such context-based predictions and results showed that ConF predicts object properties from global image appearance more accurately than kNN, while being more memory efficient and much faster.

We also used ConF to speed-up and improve object class detection (with DPM and EE-SVM). We first showed how ConF can be employed to dynamically select a small subset of model components which is most relevant for a test image. Running only the selected components led to a speed-up ($2\times$ for DPM and $10\times$ for EE-SVM). Interestingly, in some cases we gained an improvement in accuracy as well, by not running the components that led to false detections. Then, we trained a second ConF to predict at which positions and scales objects are likely to appear in a given test image. By incorporating this location information in the detector score, we reduced the false positive rate by removing detections in unlikely locations and improved detection performance (AP +1-2%). Finally, we trained a third ConF to predict what classes are present in each image, and run only the corresponding detectors. This greatly reduced the number of detectors run ($20\times$), and removed some false-positives, achieving an improvement of $+4.8\%$.

Acknowledgement. We gratefully acknowledges support by SNSF fellowship PBEZP-2142889 and ERC Starting Grant VisCul.

References

[1] University of amsterdam and euvision technologies at ilsvrc2013 (ILSVRC). http://www.image-net.org/challenges/LSVRC/2013/slides/ILSVRC2013-UvA-Euvision-web.pdf, 2013.

[2] Imagenet large scale visual recognition challenge (ILSVRC). http://www.image-net.org/challenges/LSVRC/2014/index, 2014.

[3] S. Agarwal, A. Awan, and D. Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE Trans. on PAMI*, 20(11):1475–1490, 2004.

[4] O. Aghazadeh, H. Azizpour, J. Sullivan, and S. Carlsson. Mixture component identification and learning for visual recognition. In *ECCV*, 2012.

[5] M. Aubry, D. Maturana, A. Efros, B. Russell, and J. Sivic. Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models. In *CVPR*, 2014.

[6] Y. Aytar and A. Zisserman. Enhancing exemplar svms using part level transfer regularization. In *BMVC*, 2012.

[7] H. Bay, A. Ess, T. Tuytelaars, and L. van Gool. SURF: Speeded up robust features. *CVIU*, 2008.

[8] L. Breiman. Random forests. *ML Journal*, 45(1):5–32, 2001.

[9] M. Choi, J. Lim, A. Torralba, and A. Willsky. Exploiting hierarchical context on a large database of object categories. In *CVPR*, 2010.

[10] A Criminisi, J Shotton, and E Konukoglu. Decision forests for classification, regression, density estimation, manifold learning and semi-supervised learning. *Microsoft Research Cambridge, Tech. Rep. MSRTR-2011-114*, 2011.

[11] T. Dean, M. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, and J. Yagnik. Fast, accurate detection of 100,000 object classes on a single machine. In *CVPR*, 2013.

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

[13] C Desai, D. Ramanan, and C. Folkess. Discriminative models for multi-class object layout. In *ICCV*, 2009.

[14] S. Divvala, A. Efros, and M. Hebert. How important are 'deformable parts' in the deformable parts model? In *ECCV*, 2012.

[15] P Dollár, B. Babenko, S. Belongie, P. Perona, and Z. Tu. Multiple component learning for object detection. In *ECCV*, 2008.

[16] J. Dong, W. Xia, Q. Chen, J. Feng, Z. Huang, and S. Yan. Subcategory-aware object classification. In *CVPR*, 2013.

[17] B. Drayer and T. Brox. Training deformable object models for human detection based on alignment and clustering. In *ECCV*, 2014.

[18] I. Endres, K. Shih, J. Jiaa, and D. Hoiem. Learning collections of part models for object recognition. In *CVPR*, 2013.

[19] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[20] P. Felzenszwalb, R. Girshick, and D. McAllester. Cascade Object Detection with Deformable Part Models. In *CVPR*, 2010.

[21] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Trans. on PAMI*, 32(9), 2010.

[22] R. Girshick, P. Felzenszwalb, and D. McAllester. Discriminatively trained deformable part models, release 5. http://people.cs.uchicago.edu/ rbg/latent-release5/, 2012.

[23] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[24] P. Gronat, G. Obozinski, J. Sivic, and T. Pajdla. Learning and calibrating per-location classifiers for visual place recognition. In *CVPR*, 2013.

[25] C. Gu and X. Ren. Discriminative mixture-of-templates for viewpoint classification. In *ECCV*, 2010.

[26] C. Gu, P. Arbeláez, Y. Lin, K. Yu, and J. Malik. Multi-component models for object detection. In *ECCV*, 2012.

[27] H. Harzallah, F. Jurie, and C. Schmid. Combining efficient object localization and image classification. In *ICCV*, 2009.

[28] G. Heitz and D. Koller. Learning spatial context: Using stuff to find things. In *ECCV*, 2008.

[29] Y. Jia. Caffe: An open source convolutional architecture for fast feature embedding. http://caffe.berkeleyvision.org/, 2013.

[30] M. Juneja, A. Vedaldi, CV Jawahar, and A. Zisserman. Blocks that shout: Distinctive parts for scene classification. In *CVPR*, 2013.

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[32] S. Lazebnik, C. Schmid, and J Ponce. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *CVPR*, 2006.

[33] C. Liu, J. Yuen, and A. Torralba. Nonparametric scene parsing: Label transfer via dense scene alignment. In *CVPR*, 2009.

[34] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[35] T. Malisiewicz. Ensemble of exemplar-svms, implementation. https://github.com/quantombone/exemplarsvm, 2011.

[36] T. Malisiewicz, A. Gupta, and A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *ICCV*, 2011.

[37] D. Modolo, A. Vezhnevets, O. Russakovsky, and V. Ferrari. Joint calibration of ensemble of exemplar svms. In *CVPR*, 2015.

[38] F. Moosman, B. Triggs, and F. Jurie. Fast discriminative visual codebook using randomized clustering forests. In *NIPS*, 2006.

[39] K. Murphy, A. Torralba, and W. T. Freeman. Using the forest to see the trees: A graphical model relating features, objects, and scenes. In *NIPS*, 2003.

[40] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *IJCV*, 42(3):145–175, 2001.

[41] E. Parzen. On the estimation of a probability density function. *Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

[42] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in context. In *ICCV*, 2007.

[43] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 2015. doi: 10.1007/s11263-015-0816-y.

[44] B. Russel and A. Torralba. LabelMe: a database and web-based tool for image annotation. *IJCV*, 77(1-3):157–173, 2008.

[45] B. Russell, A. Torralba, C. Liu, R. Ferugs, and W. Freeman. Object recognition by scene alignment. In *NIPS*, 2007.

[46] M.A. Sadeghi and D. Forsyth. 30hz object detection with dpm v5. In *ECCV*, 2014.

[47] A. Shrivastava, T. Malisiewicz, A. Gupta, and A. Efros. Data-driven visual similarity for cross-domain image matching. In *SIGGRAPH*, 2011.

[48] S. Singh, A. Gupta, and A. Efros. Unsupervised discovery of mid-level discriminative patches. In *ECCV*, 2012.

[49] H. Song, S. Zickler, T. Althoff, R. Girshick, M. Fritz, C. Geyer, P. Felzenszwalb, and T. Darrell. Sparselet models for efficient multiclass object detection. In *ECCV*, 2012.

[50] S. Song and J. Xiao. Sliding shapes for 3d object detection in depth images. In *ECCV*, 2014.

[51] Z. Song, Q. Chen, Z. Huang, Y. Hua, and S. Yan. Contextualizing object detection and classification. In *CVPR*, 2011.

[52] M. Sun and S. Savarese. Articulated part-based model for joint object detection and pose estimation. In *ICCV*, 2011.

[53] J. Tighe and S. Lazebnik. Superparsing: Scalable nonparametric image parsing with superpixels. In *ECCV*, 2010.

[54] J. Tighe and S. Lazebnik. Finding things: Image parsing with regions and per-exemplar detectors. In *CVPR*, 2013.

[55] A. Torralba. Contextual priming for object detection. *IJCV*, 53(2):153–167, 2003.

[56] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *IJCV*, 104(2):154–171, 2013.

[57] A. Vezhnevets and V. Ferrari. Associative embeddings for large-scale knowledge transfer with self-assessment. In *CVPR*, 2014.

[58] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba. SUN database: Large-scale scene recognition from Abbey to Zoo. In *CVPR*, 2010.

[59] J. Yang, B. Price, S. Cohen, and Y. Ming-Hsuan. Context driven scene parsing with attention to rare classes. In *CVPR*, 2014.

[60] X. Zhu, C. Vondrick, D. Ramanan, and C. Fowlkes. Do we need more training data or better models for object detection? In *BMVC*, 2012.